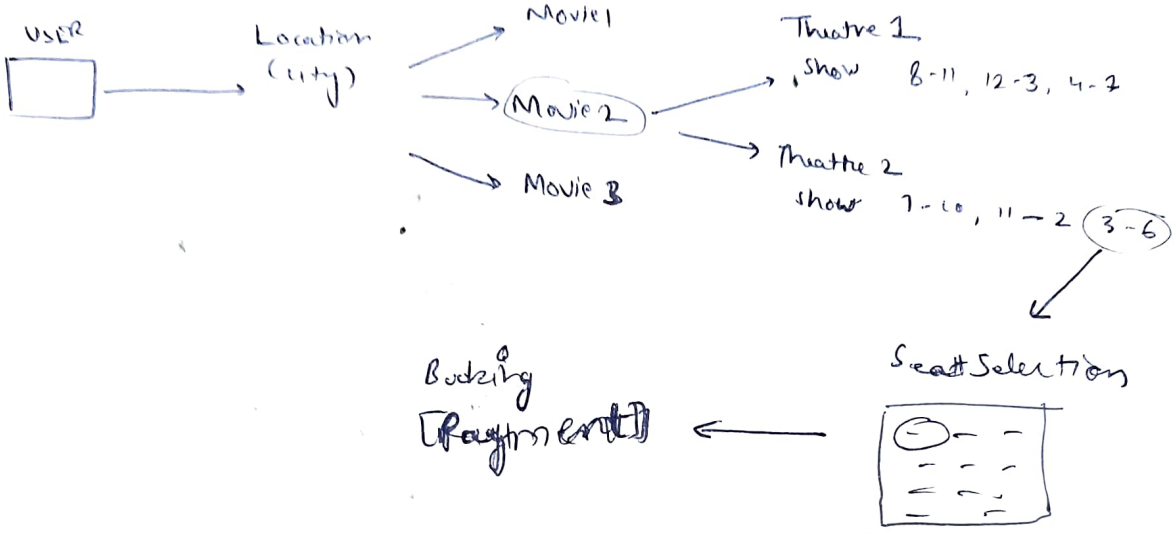
Design Movie Ticket Booking Application

Important follow up question : Concurrency Management.
How to make sure same seat is not assigned to same person.

Design + Concurrency + Code

STEP-I Rough Flow to gather requirements and identify Objects



STEP II Objects :

1. User
2. Movie
3. City
4. Theatre
5. Screen
6. Shows
7. Seats
8. Booking
9. Payment

↳ These are the standard objects which we can see as of now.

Using Top to down Approach for designing here.

5
Theatre Controller
Map<City, List<Theatre>>
cityVsTheatre;
List<Theatre> allTheatre;
3
<ENUM>
City
Bangalore,
Delhi
4
Theatre
String id;
String address;
City city;
List<Screen> screens;
List<Show> shows;
6
Screen
String id;
List<Seat> seats;
1
Movie
String id;
String name;
int duration;
2
Movie Controller
Map<City, List<Movie>>
cityVsMovie;
List<Movie> allMovies;
7
Show
String showId;
Movie movie;
Screen screen;
int startTime;
List<String> bookedSeatsId;
8
Seat
String id;
int row;
SeatCategory category;
9
<ENUM>
SeatCategory
Silver
Gold
Platinum
10
Booking
Show show;
List<Seats> seats;
Payment payment;
11
Payment
String paymentId;

How to manage concurrency?

① USER 1 → Seat [30] ← USER 2

Two person cannot select same seat

② USER 1 → Seat [30] : If user 1 has selected seat 1, it should be locked for 10 minutes. Even if the payment fails, it should be locked.

Concurrency handled in two ways

Pessimistic Lock

⇓

It will lock when reading.

Let's say User1 is Reading resource [R1]. It will lock R1 resource. User 2 will not be able to read it

Lock
User1 → [Read] → Update → Lock Release

User 2 → [Read] X

Optimistic Lock

⇓

It allows multiple users to read.

V1 → Read (version 1)

V2 → Read (version 1) → [Update] (check)

↓

Lock

↓

update resource & version update

↓

release Lock

So before locking & updating it will read if the current version of resource is same when it was read.

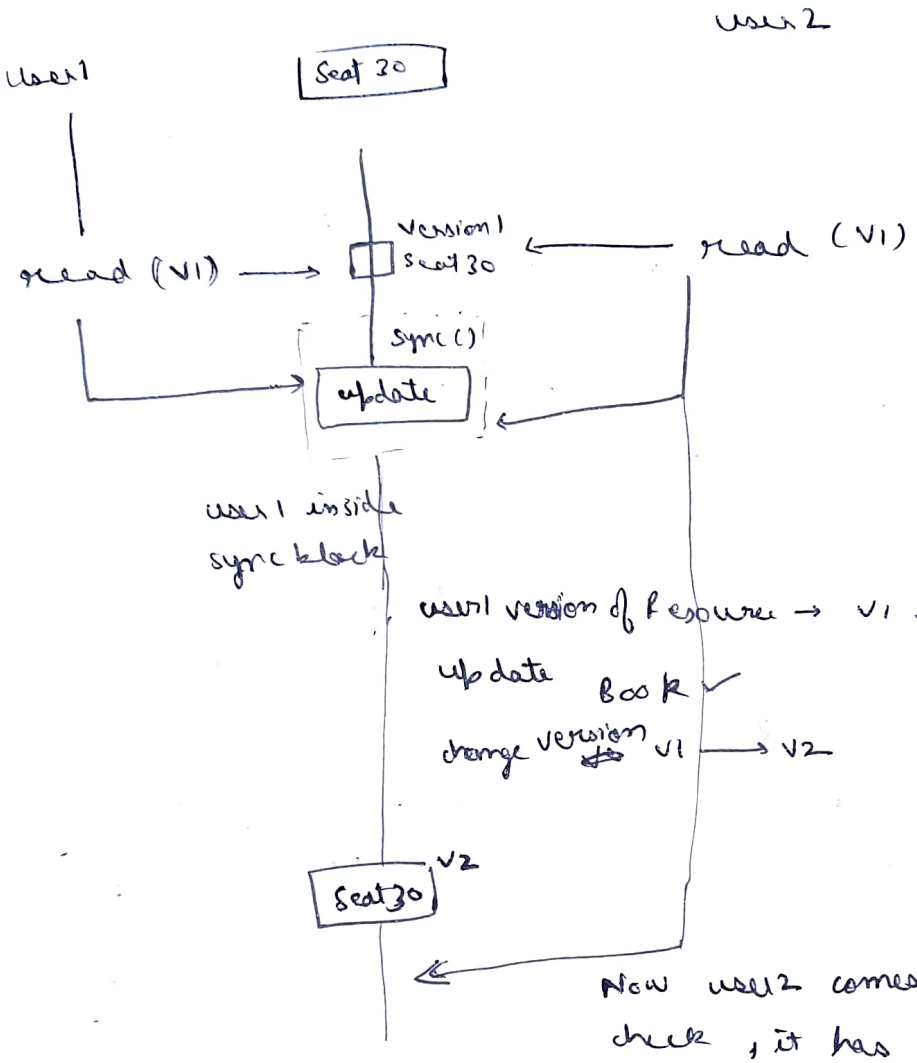
How to lock resource for some time?

Using Amazon Redis : In this we can set expiry time for lock.

In case of Book my show example →

We will not use pessimistic lock → as multiple users will not be able to see available seats at same time.

Optimistic Lock →



We will use Amazon Redios to acquire lock for 10 minutes

As soon as user1 locks seat30, it will be locked for 30 minutes even if payment fails.